**Exercise 5 : Task Management System**

**Explain the different types of linked lists (Singly Linked List, Doubly Linked List).**

**Singly Linked List:**

* Each node contains a data element and a reference (or pointer) to the next node in the list.
* We can only traverse the list in one direction (forward), from the head node to the end.
* Insertion and deletion are relatively efficient (O(1) at the head if you have the reference).

**Doubly Linked List:**

* Each node contains a data element, a reference to the next node, and a reference to the previous node.
* You can traverse the list in both directions (forward and backward).
* Allows more efficient deletion of nodes (O(1)) if you have a reference to the node to be deleted, but requires more memory due to the extra reference.

**Analyze the time complexity of each operation.**

**Add Operation:**

* **Complexity:** O(1)
* Adding a task at the head of the list involves creating a new node and updating the head reference, which is done in constant time.

**Search Operation:**

* **Complexity:** O(n)
* Searching requires traversing the list from the head to the end, so in the worst case, you may have to look through all nodes.

**Traverse Operation:**

* **Complexity:** O(n)
* Traversing the list involves visiting each node once, so it is linear in time complexity.

**Delete Operation:**

* **Complexity:** O(n)
* Deleting a task involves finding it first (O(n)) and then adjusting the pointers to remove it from the list.

**Discuss the advantages of linked lists over arrays for dynamic data.**

* Linked lists offer significant advantages over arrays for dynamic data by providing a flexible structure that can grow or shrink in size without the need for pre-allocation, unlike arrays which have a fixed size.
* They allow for efficient insertions and deletions, particularly when operations occur at the list's head or a known node, avoiding the costly element shifting required in arrays.
* Additionally, linked lists only allocate memory as needed for each element, reducing memory wastage compared to the fixed block allocation required for arrays.
* These features make linked lists well-suited for situations where the data size and structure are frequently changing.